WHAT IS CLAIMED IS:

1. A computer-readable medium having stored thereon a data structure representing a linked list in a hash table, comprising:

a key field providing an identifier for data stored in a node in the linked list;

a protected pointer field comprising a pointer that functions to indicate that a node is part of the linked list when the pointer points to the node and the protected pointer field is in the linked list; and

a reference counter field comprising a reference counter that indicates the number of references currently made to a node, such that the reference counter must be zero and none of the protected pointers in the linked list can be pointing to the node before the node can be destroyed.

2. The computer-readable medium of claim 1 wherein the protected pointer field further comprises a counter that is changed whenever the pointer is changed.

3. The computer-readable medium of claim 1 wherein the reference counter field further comprises

a status indicator field that indicates the current status of the node.

4. The computer-readable medium of claim 3 wherein the status indicator field indicates one of a status of live, deleted, dead, or unlinked.

5. The computer-readable medium of claim 3 wherein the reference counter field further comprises a generation counter field that is incremented each time the node is destroyed.

6. The computer-readable medium of claim 1 further comprising a protected pointer field within a node that points to another node in the linked list.

7. The computer-readable medium of claim 6 wherein the protected pointer field in a node further comprises a marking field that indicates whether the node should be removed from the linked list before the node is actually removed from the linked list.

8. The computer-readable medium of claim 7 wherein the marking field is not set to indicate that the node should be removed from the linked list before the node is removed from the linked list if the node is the last node in the linked list.

9. The computer-readable medium of claim 1 further comprising a hash signature field associated

with a node in the linked list, wherein the hash signature field comprises a hash signature formed from the key in the key field.

10. The computer-readable medium of claim 9 wherein the reference counter for a node is incremented before comparing the key in the key field to a search key.

11. The computer-readable medium of claim 10 wherein the reference counter for a node is not incremented before comparing the hash signature in the hash signature field to a search hash signature.

12. The computer-readable medium of claim 10 wherein the key is an object having an object reference count and wherein the reference counter supercedes the object reference count such that the object reference count does not require modification when a reference to the key object is issued.

13. A method of searching a hash table to find a node containing a search key, the method comprising:

using the search key to form a search hash signature;

using the search hash signature to locate the beginning of a linked list of nodes;

traversing through the linked list of nodes from the beginning of the linked list, traversing comprising at each node:

comparing the search hash signature to a stored hash signature stored in the node;

if the search hash signature does not match the stored hash signature, traversing to the next node in the linked list;

if the search hash signature matches the stored hash signature, comparing the search key to a stored key stored in the node;

if the search key matches the stored key, returning a pointer to the node; and

if the search key does not match the stored key, traversing to the next node in the linked list.

14. The method of claim 13 further comprising incrementing a reference counter stored in the node before comparing the search key to the key stored in the node.

15. The method of claim 13 wherein traversing to a next node comprises using a protected pointer stored in a node, wherein the protected pointer comprises a counter and a pointer to the next node.

16. The method of claim 13 wherein traversing further comprises examining an unlink node field in the node before comparing the search hash signature to the stored hash signature to determine if the node has been marked for unlinking from the linked list.

17. The method of claim 16 wherein traversing further comprises removing the node from the linked list if it has been marked for unlinking.

18. The method of claim 13 wherein traversing further comprises examining a status field stored in the node to determine if the node has been deleted.

19. The method of claim 18 wherein traversing further comprises marking a node for unlinking if the status field indicates the node is deleted and the node is not marked for unlinking.

20. The method of claim 13 wherein traversing further comprises examining a status field stored in the node to determine if the node is dead.

21. The method of claim 20 wherein traversing further comprises marking the node for unlinking if the status field indicates the node is dead and the node is not marked for unlinking.

22. The method of claim 13 wherein traversing further comprises before comparing keys, determining if a protected pointer comprising a pointer that was followed to traverse to the node, has changed since traversing to the node.

23. The method of claim 22 wherein the protected pointer further comprises a counter.

24. The method of claim 23 wherein if the protected pointer has changed, traversing further comprises restarting the traversal at the beginning of the linked list.

25. The method of claim 24 wherein the change in the protected pointer comprises a change in the counter.

26. The method of claim 13 further comprising using the returned pointer to the node to change a status of the node to deleted.

27. The method of claim 26 further comprising reducing a reference counter stored in the node by one when changing the status of the node to deleted.

28. The method of claim 27 further comprising determining if this is the last node in the linked list and unlinking the node from the list if it is

the last node in the list and marking the node for unlinking if it is not the last node in the list.

29. The method of claim 27 further comprising marking the node for unlinking after changing the status of the node to deleted.

30. The method of claim 29 further comprising unlinking the node from the linked list.

31. The method of claim 29 further comprising examining a reference counter to determine if the node is still being referenced and destroying the node if it is no longer being referenced.